CUPS Software Administrators Manual

CUPS-SAM-1.0.0

Easy Software Products

Table of Contents

Table of Contents

Preface

This software administrators manual provides printer administration information for the Common UNIX Printing System ("CUPS") Version 1.0.0.

System Overview

The Common UNIX Printing System provides a portable printing layer for UNIX® operating systems. It has been developed by Easy Software Products to promote a standard printing solution for all UNIX vendors and users. CUPS provides the System V and Berkeley command-line interfaces.

CUPS uses the Internet Printing Protocol (IETF-IPP) as the basis for managing print jobs and queues. The Line Printer Daemon (LPD, RFC1179), Server Message Block (SMB), and AppSocket protocols are also supported with reduced functionality.

CUPS adds network printer browsing and PostScript Printer Description ("PPD")-based printing options to support real world applications under UNIX.

CUPS also includes a customized version of GNU GhostScript (currently based off GNU GhostScript 4.03) and an image file RIP that can be used to support non-PostScript printers.

Document Overview

This software administrators manual is organized into the following sections:

- 1 - Printing System Overview
- 2 - Building and Installing CUPS
- 3 - Printer Queue Management
- 4 - Printing System Management
- 5 - Printer Accounting

1 - Printing System Overview

This chapter provides an overview of how the Common UNIX Printing System works.

The Printing Problem

For years *the printing problem* has plagued UNIX®. Unlike Microsoft® Windows® or MacOS, UNIX has no standard interface or system in place for supporting printers. Among the solutions previously available, the Berkeley and System V printing systems are the most prevalent.

These printing systems support line printers (text only) or PostScript printers (text and graphics), and with some coaxing they can be made to support a full range of printers and file formats. However, because each varient of the UNIX operating system uses a different printing system than the next, developing printer drivers for a wide range of printers is extremely difficult. That combined with the limited volume of customers for each UNIX varient has forced most printer vendors to give up supporting UNIX entirely.

The Common UNIX Printing System, or CUPS, is designed to eliminate *the printing problem*. One common printing system can be used by all UNIX varients to support the printing needs of users. Printer vendors can use its modular filter interface to develop a single driver program that supports a wide range of file formats with little or no effort. Since CUPS provides both the System V and Berkeley printing commands, users (and applications) can reap the benefits of this new technology with no changes.

The Technology

CUPS is based upon an emerging Internet standard called the Internet Printing Protocol, or IPP. IPP has been embraced by dozens of printer and printer server manufacturers, and will be supported by the next Microsoft Windows operating system.

IPP defines a standard protocol for printing as well as managing print jobs and printer options like media size, resolution, and so forth. Like all IP-based protocols, IPP can be used locally or over the Internet to printers hundreds or thousands of miles away. Unlike other protocols, however, IPP also supports access control, authentication, and encryption, making it a much more secure printing solution than older ones.

IPP is layered on top of the Hyper-Text Transport Protocol, or HTTP, which is the basis of web servers on the Internet. This allows the user to view documentation and status information on a printer or server using their web browser.

CUPS provides a complete IPP/1.0-based printing system that provides Basic authentication and domain or IP-based access control. Digest authentication and TLS encryption will be available in future versions of CUPS.

Jobs

Each file that is submitted for printing is called a *job*. Jobs are identified by a unique number starting at 1 and are assigned to a particular destination (usually a printer). Jobs can also have options associated with them such as media size, number of copies, and priority.

Classes

CUPS supports collections of printers known as *classes*. Jobs sent to a class are forwarded to the first available printer in the class.

Filters

Filters allow a user or application to print many types of files without extra effort. Print jobs sent to a CUPS server are filtered before sending them to a printer. Some filters convert job files to different formats that the printer can understand. Others perform page selection and ordering tasks. *Backend* filters perform the most important task of all - they send the filtered print data to the printer.

CUPS provides filters for printing many types of image files, HP-GL/2 files, PDF files, and text files. CUPS also supplies PostScript and image file Raster Image Processors, or RIPs, that convert PostScript or image files into bitmaps that can be sent to a raster printer.

CUPS provides backends for printing over parallel and serial ports, and over the network via the JetDirect (AppSocket), Server Message Block, and Line Printer Daemon protocols.

Printer Drivers

Printer drivers in CUPS consist of one of more filters specific to a printer. CUPS includes a sample printer driver for Hewlett-Packard LaserJet and DeskJet printers. While this driver does not generate optimal output for different printer models, it does demonstrate how you can write your own printer drivers and incorporate them into CUPS.

Networking

Printers and classes on the local system are automatically shared with other systems on the network. This allows you to setup one system to print to a printer and use this system as a printer server or spool host for all of the others. If there is only one occurrence of a printer on a network, then that printer can be accessed using its name alone. If more than one printer exists with the same name, users must select the printer by specifying which server to use (e.g. "printer@host1" or "printer@host2".)

CUPS also provides *implicit classes*, which are collections of printers and/or classes with the same name. This allows you to setup multiple servers pointing to the same physical network printer, for example, so that you aren't relying on a single system for printing. Because this also works with printer classes, you can setup multiple servers and printers and never worry about a "single point of failure" unless all of the printers and servers goes down!

2 - Building and Installing CUPS

This chapter shows how to build and install the Common UNIX Printing System. If you are installing a binary distribution from the CUPS web site, proceed to the section titled, Installing a Binary Distribution.

Installing a Source Distribution

Requirements

You'll need an ANSI C compiler to build CUPS on your system. As its name implies, CUPS is designed to run on the UNIX operating system, however the CUPS interface library and most of the filters and backends supplied with CUPS should also run under Microsoft® Windows®.

For the image file filters and PostScript RIP, you'll need the JPEG, PNG, TIFF, and ZLIB libraries. CUPS will build without these, but with reduced functionality. Easy Software Products maintains a mirror of the current versions of these libraries at:

ftp://ftp.easysw.com/pub/libraries

If you make changes to the man pages you'll need GNU groff or another nroff-like package. GNU groff is available from:

ftp://ftp.gnu.org/pub/groff

The documentation is formatted using the HTMLDOC software. If you need to make changes you can get the HTMLDOC software from:

http://www.easysw.com/htmldoc

Compiling CUPS

CUPS uses GNU autoconf to configure the makefiles and source code for your system. To configure CUPS for your system type:

```
% ./configure ENTER
```

The default installation will put the CUPS software in the `/usr` and `/var` directories on your system, which will overwrite any existing printing commands on your system. To install the CUPS software in another location use the `--prefix` option:

```
% ./configure --prefix=/usr/local ENTER
```

If the PNG, JPEG, TIFF, and ZLIB libraries are not installed in a system default location (typically `/usr/include` and `/usr/lib`) you'll need to set the `CFLAGS` and `LDFLAGS` environment variables prior to running configure:

```
% setenv CFLAGS "-I/some/directory"
% setenv LDFLAGS "-L/some/directory"
% ./configure ... ENTER
```

Once you have configured things, just type:

```
% make ENTER
```

to build the software.

Installing the Software

To install the software type:

```
% make install ENTER
```

Running the Software

Once you have installed the software you can start the CUPS daemon by typing:

```
% /usr/sbin/cupsd & ENTER
```

Installing a Binary Distribution

We are currently distributing CUPS binary distributions in TAR format with installation and removal scripts.

> **WARNING:**
>
> Installing CUPS will overwrite your existing printing system. If you experience difficulties with the CUPS software and need to go back to your old printing system, you will need to remove the CUPS software with the provided script and reinstall the printing system from your operating system CDs.

To install the CUPS software you will need to be logged in as root (doing an "su" is good enough). Once you are the root user, run the installation script with:

```
./cups.install ENTER
```

After asking you a few yes/no questions the CUPS software will be installed and the scheduler will be started automatically.

3 - Printer Queue Management

This chapter discusses how to add, modify, and delete print queues on your system.

The lpadmin Command

The `lpadmin` command allows you to perform most printer administration tasks from the command-line. Since `lpadmin` is also a System V printing system command, it is located in the `/usr/lib` directory instead of a more common one like `/usr/bin` or `/usr/sbin`.

Adding and Modifying Printers

To add a printer to CUPS you simply run the `lpadmin` command with the "-p" option:

```
% /usr/lib/lpadmin -pprinter -E -vdevice -Pppd ENTER
```

Spaces between the option letter and value are optional.

The *printer* name can be up to 127 letters, digits, hyphens, and underscores. Unlike other printing systems, the printer name in CUPS is *not* case-sensitive, so you can't add two printers named `LaserJet` and `laserjet`.

The *device* argument specifies the device URI or filename for the printer. The following devices are supported in a basic installation of CUPS:

file:/dev/filename

/dev/filename

Sends all output to the specified file.

http://[username:password@]hostname[:port]/resource

ipp://[username:password@]hostname[:port]/resource

Sends all output to the specified IPP printer or server. The *port* parameters defaults to 631.

lpd://hostname/queue

Sends all output to the specified LPD printer queue.

parallel:/dev/filename

Sends all output to the specified parallel port device.

serial:/dev/filename[?options]

Sends all output to the specified serial port device. The *options* can be any of the following separated by the plus (+) character:
`baud=`*`rate`* - Sets the baud rate for the device.
`bits=`*`7 or 8`* - Sets the number of data bits.
`parity=`*`even`* - Sets even parity checking.
`parity=`*`odd`* - Sets odd parity checking.
`parity=`*`none`* - Turns parity checking off.

smb://[username:password@]hostname/queue

Sends all output to the specified SMB (Windows) printer queue using the SAMBA software.

socket://hostname[:port]

Sends all output to the specified printer using the AppSocket protocol. The *port* parameter defaults to 9100.

The *ppd* argument specifies the PostScript Printer Description file to use for this printer. Many options (such as media size, etc.) will not be available if you omit this part of the `lpadmin` command.

Using Standard Printer Drivers

The `lpadmin` command allows you to use "standard" PPD files and interface scripts located in the

`/usr/share/cups/model` directory with the "-m" option:

```
% /usr/lib/lpadmin -pprinter -E -vdevice -mmodel ENTER
```

The *model* argument specifies the name of the PPD file or interface script. For example, to add a printer using the sample HP DeskJet series driver connected to parallel port 1 under Linux you would use:

```
% /usr/lib/lpadmin -pDeskJet -E -vparallel:/dev/par1 -mdeskjet.ppd ENTER
```

Removing Printers

To remove a printer to CUPS you simply run the `lpadmin` command with the "-x" option:

```
% /usr/lib/lpadmin -xprinter ENTER
```

Printer Classes

CUPS allows you to group similar printers in a *printer class*. When a user sends a print job to a class, the job will be processed by the first available printer in that class.

To add a printer to a class you simply run the `lpadmin` command with the "-p" and "-c" options:

```
% /usr/lib/lpadmin -pprinter -cclass ENTER
```

The *class* is created automatically if it doesn't exist. To remove a class just use the "-x" option:

```
% /usr/lib/lpadmin -xclass ENTER
```

Setting the Default Printer

To set the default printer or class simply run the `lpadmin` command with the "-d" option:

```
% /usr/lib/lpadmin -ddestination ENTER
```

The *destination* argument is the name of the printer or class.

Starting and Stopping Printers

The `enable` and `disable` commands start and stop printer queues, respectively:

```
% /usr/bin/enable printer ENTER
```

```
% /usr/bin/disable printer ENTER
```

Printers that are disabled may still accept jobs for printing, but won't actually print any files until they are restarted. This is useful if the printer malfunctions and you need time to correct the problem. Any queues jobs are printed after the printer is enabled (started).

Accepting and Rejecting Print Jobs

The `accept` and `reject` commands accept and reject print jobs for the named printer, respectively:

```
% /usr/lib/accept printer ENTER
% /usr/lib/reject printer ENTER
```

As noted above, a printer can be stopped but accepting new print jobs. A printer can also be rejecting new print jobs while it finishes those that have been queued. This is useful for when you must perform maintenance on the printer and will not have it available to users for a long period of time.

4 - Printing System Management

This chapter shows how you can configure the CUPS server.

Changing the Configuration Files

All of the server configuration files are located in the `/var/cups/conf` directory. Once you have made a change to a file you need to restart the CUPS server by sending it a HUP signal or using the supplied script "`cups.sh`":

```
% ./cups.sh restart ENTER
```

The binary distribution installs the script in the `init.d` directory with the name `lp` or `lpd` depending on the vendor-supplied printing system.

Temporary Files

Normally CUPS puts all of its temporary files in `/var/tmp`. If you'd like to change this directory you'll need to edit the `/var/cups/conf/cupsd.conf` file.

Start by creating the new temporary directory and setting the appropriate permissions:

```
% mkdir /foo/bar/tmp ENTER
% chmod a+rwxt /foo/bar/tmp ENTER
```

Then change the line containing the `TempDir` directive in the `cupsd.conf` to the directory that you've created:

```
TempDir /foo/bar/tmp
```

Finally, restart the server as outlined in the first section of this chapter.

Network Configuration

The default configuration of the CUPS server listens for connections from all network interfaces on port 631 (the standard IPP port). Administration functions are limited to local connections with the appropriate username and password.

If you'd like to limit access to your system you'll need to edit the `/var/cups/conf/cupsd.conf` file.

Port

The `Port` directive specifies a port to listen on for all interfaces. Besides the standard IPP port (631) you can also setup your server to listen on the HTTP port (80) to use your CUPS server as a standard web server as well.

Listen

The `Listen` directive specifies a listening address and port, extending the functionality of the `Port` directive. If you want to allow connections only from the local machine you can use:

```
Listen 127.0.0.1:631
```

instead of the `Port` directive.

If you want to limit access to a specific network/subnet, make sure you specify only the network address and not your system's network address!

BrowsePort

The `BrowsePort` directive controls which port is monitored for remote printers. By default it is set to the IPP port (631), however you can change it as needed.

NOTE:

You must set the `BrowsePort` to the same value on all of the systems that you want to see.

BrowseAddress

The `BrowseAddress` directive specifies a broadcast address to use when sending printer status updates over the network. The default browse address is `255.255.255.255` which will send printer information to all subnets.

NOTE:

If you are using HP-UX 10.20 and a subnet that is not 24, 16, or 8 bits, printer browsing (and in fact all broadcast reception) will not work. This problem appears to be fixed in HP-UX 11.0.

Printer Security

CUPS provides IP and domain-name based access control and Basic authentication for authentication.

Location

The `Location` directive defines access control for a specific HTTP directory. The following pseudo directories are provided by the CUPS server:

- `/admin` - This is the URI that must be referenced to do printer administation commands.
- `/classes` - This is the URI that must be referenced to access printer classes.
- `/jobs` - This is the URI that must be referenced to access jobs.
- `/printers` - This is the URI that must be referenced to access printers.

All other directories are taken from the `/usr/share/cups/doc` directory.

The `Location` directive surrounds the other access control directives described below. The default server configuration uses:

```
<Location /admin>
AuthType Basic
AuthClass System

Order Deny,Allow
Deny From All
Allow From 127.0.0.1
</Location>
```

Order

The `Order` directive defines the default access control. The following values are supported:

- `Order Allow,Deny` - Allow requests from all systems *except* for those listed in a `Deny` directive.
- `Order Deny,Allow` - Allow requests only from those listed in an `Allow` directive.

The `Order` directive must appear inside a `Location` directive.

Allow

The `Allow` directive specifies a hostname, IP address, or network that is allowed access to the server:

```
Allow from All
Allow from None
Allow from *.domain.com
Allow from .domain.com
Allow from host.domain.com
Allow from nnn.*
Allow from nnn.nnn.*
Allow from nnn.nnn.nnn.*
Allow from nnn.nnn.nnn.nnn
Allow from nnn.nnn.nnn.nnn/mm
Allow from nnn.nnn.nnn.nnn/mmm.mmm.mmm.mmm
```

`Allow` directives are cummulative, so multiple `Allow` directives can be used to allow access for multiple hosts or networks. The `/mm` notation specifies a CIDR netmask:

mm	netmask
0	0.0.0.0
1	128.0.0.0
2	192.0.0.0
...	...
8	255.0.0.0
16	255.255.0.0
24	255.255.255.0
32	255.255.255.255

The `Allow` directive must appear inside a `Location` directive.

Deny

The `Deny` directive specifies a hostname, IP address, or network that is allowed access to the server:

```
Deny from All
Deny from None
Deny from *.domain.com
Deny from .domain.com
Deny from host.domain.com
Deny from nnn.*
Deny from nnn.nnn.*
Deny from nnn.nnn.nnn.*
Deny from nnn.nnn.nnn.nnn
Deny from nnn.nnn.nnn.nnn/mm
Deny from nnn.nnn.nnn.nnn/mmm.mmm.mmm.mmm
```

`Deny` directives are cummulative, so multiple `Deny` directives can be used to allow access for multiple hosts or networks. The `/mm` notation specifies a CIDR netmask:

mm	netmask
0	0.0.0.0
1	128.0.0.0
2	192.0.0.0
...	...
8	255.0.0.0
16	255.255.0.0
24	255.255.255.0
32	255.255.255.255

The `Deny` directive must appear inside a `Location` directive.

AuthType

The `AuthType` directive defines the type of authentication to perform:

- `None` - No authentication should be performed (default.)
- `Basic` - Basic authentication should be performed using the UNIX password and group files.

The `AuthType` directive must appear inside a `Location` directive.

AuthClass

The `AuthClass` directive defines what level of `Basic` access is required:

- `Anonymous` - No authentication should be performed (default.)
- `User` - A valid username and password is required.
- `System` - A valid username and password is required, and the username must belong to the "sys" group (this can be changed using the `SystemGroup` directive, below.
- `Group` - A valid username and password is required, and the username must belong to the group named by the `AuthGroupName` directive.

The `AuthClass` directive must appear inside a `Location` directive.

AuthGroupName

The `AuthGroupName` directive sets the group to use for `Group` authentication.

The `AuthGroupName` directive must appear inside a `Location` directive.

SystemGroup

The `SystemGroup` directive sets the administration group used when authenticating the `System` type. It defaults to the "sys" group.

File Formats

CUPS provides a MIME-based file typing and filtering mechanism to convert files to a printable format for each printer. The `mime.types` and `mime.convs` files define the file type and filters that are available on the system.

mime.types

The `mime.types` defines the known file types. Each line of the file starts with the MIME type and may be followed by one or more file type recognition rules. For example, the `text/html` file type is defined as:

```
text/html html htm \
          printable(0,1024) + (string(0,"<HTML>") string(0,"<!DOCTYPE"))
```

The first two rules say that any file with an extension of ".html" or ".htm" is a HTML file. The third rules says that any file whose first 1024 characters are printable text and starts with the strings "<HTML>" or "<!DOCTYPE" is a HTML file as well.

The first two rules deal solely with the name of the file being typed. This is useful when the original filename is known, however for print files the server doesn't always have a filename to work with. The third rule takes care of this possibility and automatically figures out the file type based upon the contents of the file instead.

The available tests are:

- `( expr )` - Parenthesis for expression grouping
- `+` - Logical AND
- `,` or whitespace - Logical OR
- `!` - Logical NOT
- `match("pattern")` - Pattern match on filename
- `extension` - Pattern match on "*.extension"
- `ascii(offset,length)` - True if bytes are valid printable ASCII (CR, NL, TAB, BS, 32-126)
- `printable(offset,length)` - True if bytes are printable 8-bit chars (CR, NL, TAB, BS, 32-126, 160-254)
- `string(offset,"string")` - True if bytes are identical to string
- `char(offset,value)` - True if byte is identical
- `short(offset,value)` - True if 16-bit integer is identical (network or "big-endian" byte order)
- `int(offset,value)` - True if 32-bit integer is identical (network or "big-endian" byte order)
- `locale("string")` - True if current locale matches string

mime.convs

The `mime.convs` file defines all of the filter programs that are known to the system. Each line consists of:

```
source destination cost program

text/plain application/postscript 50 texttops
application/vnd.cups-postscript application/vnd.cups-raster 50 pstoraster
image/* application/vnd.cups-postscript 50 imagetops
image/* application/vnd.cups-raster 50 imagetoraster
```

The *source* field is a MIME type, optionally using a wildcard for the super-type or sub-type (e.g. "text/plain", "image/*", "*/postscript").

The *destination* field is a MIME type defined in the `mime.types` file.

The *cost* field defines a relative cost for the filtering operation from 1 to 100. The cost is used to choose between two different sets of filters when converting a file. For example, to convert from `image/jpeg` to `application/vnd.cups-raster`, you could use the `imagetops` and `pstoraster` filters for a total cost of 100, or the `imagetoraster` filter for a total cost of 50.

The *program* field defines the filter program to run; the special program "-" can be used to make two file types equivalent. The program must accept the standard filter arguments and environment variables described in the CUPS Interface Design Document:

```
program job user title options [filename]
```

If specified, the *filename* argument defines a file to read when filtering, otherwise the filter must read from the standard input. All filtered output must go to the standard output.

5 - Printer Accounting

This chapter describes the CUPS log files.

Where to Find the Log Files

The log files are normally stored in the `/var/cups/logs` directory. You can change this by editing the `/var/cups/conf/cupsd.conf` configuration file.

The access_log File

The `access_log` file lists each HTTP resource that is accessed by a web browser or CUPS/IPP client. Each line is in the so-called "Common Log Format" used by many web servers and web reporting tools:

```
host group user date-time \"method resource version\" status bytes

127.0.0.1 - - [20/May/1999:19:20:29 +0000] "POST /admin/ HTTP/1.1" 401 0
127.0.0.1 - mike [20/May/1999:19:20:31 +0000] "POST /admin/ HTTP/1.1" 200 0
```

The *host* field will normally only be an IP address unless you have changed the `HostnameLookups` directive on in the `cupsd.conf` file.

The *group* field always contains "-".

The *user* field is the authenticated username of the requesting user. If no username and password is supplied for the request then this field contains "-".

The *date-time* field is the date and time of the request in Greenwich Mean Time (a.k.a. ZULU) and is in the format:

```
[DD/MON/YYYY:HH:MM:SS +0000]
```

The *method* field is the HTTP method used ("GET", "PUT", "POST", etc.)

The *resource* field is the filename of the requested resource.

The *version* field is the HTTP specification version used by the client. For CUPS clients this will always be "HTTP/1.1".

The *status* field contains the HTTP result status of the request. Usually it is "200", but other HTTP status codes are possible. For example, 401 is the "unauthorized access" status in the example above.

The *bytes* field contains the number of bytes in the request. For POST requests the *bytes* field contains the number of bytes of non-IPP data that is received from the client.

The error_log File

The `error_log` file lists messages from the scheduler (errors, warnings, etc.):

```
level date-time message

I [20/May/1999:19:18:28 +0000] Job 1 queued on 'DeskJet' by 'mike'.
I [20/May/1999:19:21:02 +0000] Job 2 queued on 'DeskJet' by 'mike'.
I [20/May/1999:19:22:24 +0000] Job 2 was cancelled by 'mike'.
```

The *level* field contains the type of message:

- `E` - An error occurred.
- `W` - The server was unable to perform some action.
- `I` - Informational message.
- `D` - Debugging message.

The *date-time* field contains the date and time of when the page started printing. The format of this field is identical to the *data-time* field in the `access_log` file.

The *message* fields contains a free-form textual message.

The page_log File

The `page_log` file lists each page that is sent to a printer. Each line contains the following information:

```
printer user job-id date-time page-number num-copies

DeskJet root 2 [20/May/1999:19:21:05 +0000] 1 0
```

The *printer* field contains the name of the printer that printed the page. If you send a job to a printer class, this field will contain the name of the printer that was assigned the job.

The *user* field contains the name of the user (the IPP `requesting-user-name` attribute) that submitted this file for printing.

The *job-id* field contains the job number of the page being printed. Job numbers are reset to 1 whenever the CUPS server is started, so don't depend on this number being unique!

The *date-time* field contains the date and time of when the page started printing. The format of this field is identical to the *data-time* field in the `access_log` file.

The *page-number* and *num-pages* fields contain the page number and number of copies being printed of that page. For printer that can not produce copies on their own, the *num-pages* field will always be 1.